Left-incompatible Term Rewriting Systems and its Normalizing Strategy(Theory of Rewriting Systems and Its Applications)

AUTHOR(S):

Sakai, Masahiko

CITATION:

Sakai, Masahiko. Left-incompatible Term Rewriting Systems and its Normalizing Strategy(Theory of Rewriting Systems and Its Applications). 数理解析研究所講究録 1995, 918: 16-23

ISSUE DATE:

1995-08

URL:

http://hdl.handle.net/2433/59681

数理解析研究所講究録
918巻 1995年 16-23

Left-incompatible Term Rewriting Systems and its Normalizing Strategy

Masahiko Sakai,
School of Information Science, Japan Advanced Institute of Science and Technology,
Tatsunokuchi, Ishikawa, 923-12 Japan,
E-mail sakai@jaist.ac.jp

abstract: This paper extends left-incompatible term rewriting systems defined by Toyama et al.[TSEP93]. It is shown that the functional strategy is normalizing in the class, where the functional strategy is the reduction strategy which finds index by some rule selection method and top-down and left-to-right lazy pattern matching method. The reduction procedure based on the strategy is also shown.

1 Introduction

According to lazy evaluation method widely adopted in functional programming language implementations, the evaluation of an expression is postponed until the result is truly needed[Hen80, Tur85, Jon87, Dav92]. If the value of the expression is not used in anywhere, it is never evaluated. Hence, it is often observed that an expression computation which does not terminate by usual call-by-value method, called eager method, terminates by lazy method. For example, letting rules of if be

$if(true, x, y) \to x$ and
$if(false, x, y) \to y$,

$if(g(a), h(b), i(c))$ will be lazily evaluated by following steps;

(1) Select the first pattern $if(true, x, y)$.

(2) Find a position needed in order to reduce by the pattern. So $g(a)$ is selected and evaluated first.

(3) If $g(a)$ comes to $true$, reduce $if(g(a), h(b), i(c))$ to $h(b)$ and then evaluate $h(b)$. If not, select the second pattern $if(false, x, y)$ and repeat in the same way as (2) and (3).

Let the value of $g(a)$ be true and the value of $h(b)$ can be computed. The value of target expression can be computed even if the evaluation of $i(c)$ requires infinite computation steps.

The lazy evaluation is not only efficient but also effective on normalizing property as illustrated above. However, it is not so clear the class of program on which the lazy method mentioned above works as a normalizing strategy, i.e., the value of an expression can be computed if it exists.

On the other hand, Huet and Levy proposed the concept of index[HL79]. They defined a class of strong-sequential term rewriting systems (SS) in which each term not in normal form has an index and in which index reduction is normalizing. Strandh defined the class of bounded term rewriting systems (B)[Str89], which is the same class as SS shown in later[Dur94]. For efficient execution, Toyama proposed transitive system (TS), which is the subclass of SS[TSEP93]. Strandh separately defined forward-branching class[Str89], which is the same class as TS shown in later[Dur94]. Durand showed that efficient construction of matching automaton given by Huet and Levy[HL79] for simple systems, which is the proper sub-class of FB, can also be applied to FB[Dur94].

Toyama proposed left-incompatible system, called SLI in this paper, which is proper subclass of TS[TSEP93]. In the system, top-to-down and left-to-right lazy pattern matching method stated above is normalizing strategy. In this paper, we define left-incompatible system (LI) larger class

than SLI. LI is still proper sub-class of TS. We also give a reduction procedure which repeats reduction efficiently based on top-to-down and left-to-right lazy pattern matching method.

2 Term Rewriting Systems

We mainly follow the notation of [Klo92, TSEP93] and assume readers are familiar with term rewriting systems (TRS)[Klo92, DJ90]. Let $\Sigma = V \cup F$ be a signature where F and V are a set of function symbols with arity and a set of variables, respectively. T_Σ (or simply T) denotes the set of terms well constructed by symbols in Σ. A rewrite rule is a pair of terms $l \to r$ such that l is not a variable and variables in r also appear in l. A TRS R is a finite set of rewrite rules.

Letting $\Box$ be an extra constant, a term $C \in (T_{\Sigma\cup\{\Box\}} - T_\Sigma)$ is called a context denoted by $C[\,,\ldots,\,]$. For $C[\,,\ldots,\,]$ which contains n $\Box$'s and for $t_1,\ldots,t_n \in T_\Sigma$, $C[t_1,\ldots,t_n]$ denotes the obtained term by replacing $\Box$'s with $t_1,\ldots,t_n$ from left to right order.

Let σ be a substitution. We denote the application of σ to term t by $t\sigma$. Syntactical equality of terms and subterm relations are indicated by $\equiv$ and $\subseteq$ respectively.

We say that the term t reduces to s by using rule $l \to r$ (on context $C[\,]$) if and only if there exist a substitution σ and context $C[\,]$ such that $t \equiv C[l\sigma]$ and $s \equiv C[r\sigma]$. We write $t \to s$ when t reduces to s. We use $\xrightarrow{*}$ to denote reflexive and transitive closure of $\to$. $\xrightarrow{+}$ is also used to denote transitive closure of $\to$.

A TRS is called left-linear, if every variable of l occurs only once for every left-hand side l. A TRS is called ambiguous, if there exist rules $l \to r$, $l' \to r'$, term $s \subset l'$ and substitutions σ, σ' such that $l\sigma \equiv s\sigma'$ except trivial case, i.e., the rules are the same and $l \equiv s$. A left-liner and non-ambiguous TRS is orthogonal.

In this paper, we only deal with orthogonal TRS's.

3 Transitive Systems

In this section, we review transitive systems and their property according to reference [TSEP93]. The class of transitive systems is a sub-class of strongly sequential systems[HL79].

Definition 3.1 (Ω-terms) *Letting Ω be an extra constant, we represent prefixes of terms by Ω-terms in $T_{\Sigma\cup\{\Omega\}}$ (also denoted by T_Ω simply).*

- *t_Ω denotes the Ω-term obtained from a term t by replacing each variables with Ω.*
- *The prefix ordering $\succeq$ on T_Ω is defined as follows:*

$$\begin{array}{ll} t \succeq \Omega & \text{for all } t \in T_\Omega, \\ t \succeq t & \text{for each variable or constant,} \\ f(t_1,\ldots,t_n) \succeq f(s_1,\ldots,s_n) & \\ \qquad \text{if } t_i \succeq s_i \text{ for } i = 1,\ldots,n. & \end{array}$$

- *t and s are compatible, written by $t \uparrow s$, if $u \succeq t$ and $u \succeq s$ for some u; otherwise they are incompatible, denoted by $t\#s$.*

Definition 3.2 (Ω-systems) *Let R be Term Rewriting System.*

- *The set of redex schemata of R is $Red = \{l_\Omega \mid l \to r \in R\}$.*
- *The reduction relation $\to_\Omega$ (Ω-reduction) is defined on T_Ω as $C[t] \to_\Omega C[\Omega]$ where $t \uparrow s$ for some $s \in Red$ and $t \not\equiv \Omega$.*
- *NF_Ω is the set of all normal forms w.r.t. Ω-reduction.*

Lemma 3.3 ([Klo92]) *Ω-reduction is confluent and terminating.*

Definition 3.4 *The direct approximant $\omega(t)$ of an Ω-term t is the normal form of t w.r.t. Ω-reduction.*

Lemma 3.5 ([TSEP93])

- *If $t \succeq s$ then $\omega(t) \succeq \omega(s)$.*
- *Let $C[\Omega] \in NF_\Omega$. Then for all $t \in NF_\Omega$, $C[t] \in NF_\Omega$.*

Definition 3.6 *A term t is in strong head normal forms if $\omega(t) \not\equiv \Omega$.*

Lemma 3.7 ([TSEP93]) *A term t in strong head normal forms is in head normal forms.*

Definition 3.8 (Transitive index)

- *Let $C[\ ]$ be a context and let z be a fresh variable. If $z \subseteq \omega(C[z])$, the displayed occurrence of Ω in $C[\Omega]$ is called an index denoted by $C[\Omega_I]$.*
- *The displayed index in $C_1[\Omega_I]$ is transitive if for any Ω-term $C_2[\Omega_I]$, $C_2[C_1[\Omega_I]]$. The transitive index is denoted by $C_1[\Omega_{TI}]$.*

Definition 3.9 *Let R be Term Rewriting System.*

- $Red^* = \{p \mid \Omega \prec p \subseteq q$ *for some* $q \in Red\}$.
- $Red^+ = Red^* - Red$.
- *The set of preredex schemata of R is*

$$Red^{\prec} = \{p \mid \Omega \prec p \prec q \text{ for some } q \in Red\}.$$

Definition 3.10 (Transitive direction)

- *Let $Q \subseteq T_\Omega$. The displayed Ω in $C[\Omega]$ is a direction for Q if $C[z] \# Q$. A direction for Q is indicated with $C[\Omega_Q]$.*
- *Transitive direction is defined as a direction for Red^*. A transitive direction is denoted with $C[\Omega_{TD}]$.*

Lemma 3.11 ([TSEP93]) *If $C[\Omega_{TD}]$ and $C[z] \in NF_\Omega$ then $C[\Omega_{TI}]$.*

Lemma 3.12 ([TSEP93]) *Let R be a TRS. R is transitive iff every $t \in Red^{\prec}$ has a transitive direction.*

We prepare some lemmas used in later.

Definition 3.13

- *The reduction $t \to s$ on context $C[\ ]$ is called transitive reduction if $C[\Omega_{TI}]$. We write transitive reduction relation $\to_{TI}$.*
- *Let $t \equiv C[s]$, $C[\Omega_{TI}]$ and $C[\ ] \not\equiv \Box$. We write $t \supset_{TI} s$.*
- *The relation $\to_{TI} \cup \supset_{TI}$ is denoted by $\ni\!\!\!\!\to$.*

Proposition 3.14 *If $t \xrightarrow{*}_{TI} s$ and $C[\Omega_{TD}]$, then $C[t] \xrightarrow{*}_{TI} C[s]$.*

Proof Let $t \equiv C'[l\sigma]$ and $s \equiv C'[r\sigma]$ for some $l \to r \in R$, σ and $C'[\Omega_{TI}]$. Since $C[\Omega_I]$ by the definition, we have $C[C'[\Omega_{TI}]]$. This indicates $C[t] \equiv C[C'[l\sigma]] \to_{TI} C[C'[r\sigma]] \equiv C[s]$. An easy induction concludes the proof. □

Lemma 3.15 *Let R be transitive TRS. If a term t has a normal form, then there exist no infinite sequences:*

$$t \equiv t_0 \ni\!\!\!\!\to t_1 \ni\!\!\!\!\to t_2 \ni\!\!\!\!\to \cdots$$

Proof Assume infinite sequence $t_0 \ni\!\!\!\!\to t_1 \ni\!\!\!\!\to \cdots$. We can define $C_i[\ ]$ from t_i as follows.

- $C_0[\ ] \equiv \Box$. Therefore $C_0[t_0] \equiv t_0$.
- $C_{i+1}[\ ] \equiv C_i[\ ]$, if $t_i \to_{TI} t_{i+1}$.
- $C_{i+1}[\ ] \equiv C_i[C[\ ]]$, if $t_i \supset_{TI} t_{i+1}$, i.e., $t_i \equiv C[t_{i+1}]$.

We can show following properties by using proposition 3.14.

- $C_i[\Omega_I]$ for any $i \geq 0$,
- $t_i \to_{TI} t_{i+1} \Rightarrow C_i[t_i] \to_{TI} C_{i+1}[t_{i+1}]$, and
- $t_i \supset_{TI} t_{i+1} \Rightarrow C_i[t_i] \equiv C_{i+1}[t_{i+1}]$.

Since t has a normal form, there must be finite steps of $\to_{TI}$ in $t_0 \ni\!\!\!\!\to t_1 \ni\!\!\!\!\to \cdots$, otherwise we can construct infinite sequence $t_0 \equiv s_0 \to_{TI} s_1 \to_{TI} \cdots$. Therefore, there exists n such that $t_n \supset_{TI} t_{n+1} \supset_{TI} \cdots$. This is impossible. □

4 Extended Left-incompatible System and Functional Strategy

In simple left-incompatible system(SLI), an index of a given term is efficiently computed basically by a lazy pattern matching method. In this section, the author gives a left-incompatible system (LI) without losing the advantage of SLI system.

Definition 4.1 (Left incompatibility) *Let $s, t \in T_\Omega$. The left incompatibility of s and t, indicated by $t \#_{\leq} s$, is defined as follows:*

- $t \not\equiv \Omega$, $s \not\equiv \Omega$, *and*
- $f = g \Rightarrow \exists i[\forall j < i[t_j \preceq s_j] \wedge t_i \#_{\leq} s_i]$, *where* $t \equiv f(t_1, \ldots, t_n)$ *and* $s \equiv g(s_1, \ldots, s_m)$.

We write $t \#_{<} s$ *instead of* $t \#_{\leq} s \wedge \neg(s \#_{\leq} t)$. $t \#_{=} s$ *also denotes* $t \#_{\leq} s \wedge s \#_{\leq} t$.

Intuitively, $t \#_{\leq} s$ if and only if each node of t is Ω or it is the same as the corresponding node of s until the nodes are different constant or function symbols in depth-first order comparison.

Lemma 4.2 ([TSEP93]) *Let* $C[\Omega] \uparrow p$ *and let* $C[\Omega_{\{p\}}]$ *be the left most direction for* $\{p\}$. *Then*

$p \#_{\leq} q \Rightarrow C[\Omega_{\{q\}}]$.

□

Definition 4.3 ([TSEP93]) *An orthogonal TRS* (Σ, R) *is simple left-incompatible (SLI) if Red can be expressed as a list* $[p_1, \ldots, p_n]$ *satisfying the following conditions:*

- $p_i \#_{\leq} p_j$ *if* $i < j$, *and*
- $\forall p_i \in Red, q \in Red^{+}[p_i \#_{\leq} q]$.

The followings are defined by relaxed the second condition of SLI in essential. Moreover, the following definition does not require the existence of the list of *Red*.

Definition 4.4 (Left-incompatible system) *An orthogonal TRS* (Σ, R) *is left incompatible if it satisfies the following conditions:*

- $\forall p \in Red, q \in Red^{*}[p \not\equiv q \Rightarrow (p \#_{\leq} q \vee q \#_{\leq} p)]$, *and*
- $\forall q \in Red^{+}[\exists p \in Red[\neg(p \#_{\leq} q)] \Rightarrow \forall q' \in Red^{+}[q \not\equiv q' \Rightarrow (q \#_{\leq} q' \vee q' \#_{\leq} q)]]$.

Example 4.5 *Let* $Red = \{f(g(\Omega, a)), g(a, b)\}$. *The system is left-incompatible but not simple left-incompatible, since* $g(\Omega, a) \#_{<} g(a, b)$.

Definition 4.6 *A list of standard redex schemata is* $SRed = [p_1, \ldots, p_n]$ *which satisfies the following conditions:*

- $Red \subseteq SRed \subseteq Red^{*}$,
- $\forall p_i, p_j[i < j \Rightarrow p_i \#_{\leq} p_j]$, *and*
- $\forall p_i, \forall q \in (Red^{*} - SRed)[p_i \#_{\leq} q]$

Lemma 4.7 *If TRS* (Σ, R) *be left-incompatible, there exists a* $SRed = [p_1, \ldots, p_n]$. *The converse also holds.*

The lemma 4.7 is not trivial, because the left-incompatible relation $\#_{\leq}$ is not a partial order since it does not have transitivity. For example, $f(\Omega, \Omega, 0, 0) \#_{<} f(\Omega, 1, 1, 1)$ and $f(\Omega, 1, 1, 1) \#_{<} f(2, 2, \Omega, 2)$ hold, but $f(\Omega, \Omega, 0, 0) \#_{\leq} f(2, 2, \Omega, 2)$ does not hold.

Before proving this lemma, we must prepare several notions.

Definition 4.8 *The lexicographic prefix ordering* $\prec_l$ *on* T_Ω *is defined as follows.*

$$\Omega \prec_l t \quad \text{if } t \not\equiv \Omega,$$
$$f(t_1, \ldots, t_n) \prec_l f(s_1, \ldots, s_n) \quad \text{if } [t_1, \ldots, t_n] \prec\!\prec_l [s_1, \ldots, s_n].$$

where $\prec\!\prec_l$ *is lexicographic extension of* $\prec_l$.

It is easy to show that $\prec_l$ is a partial order on T_Ω.

Proposition 4.9 *Let* t *and* s *be in* T_Ω.

(a) $t \prec s \Rightarrow t \prec_l s$

(b) $t \prec_l s \Rightarrow \neg(s \#_{\leq} t)$

(c) $t \#_{<} s \Rightarrow t \prec_l s$

Proof (a) is trivial from the definitions.

For the proof of (b), we assume $t \prec_l s$. In case of $t \equiv \Omega \prec_l s \not\equiv \Omega$, we have $\neg(s \#_{\leq} t)$ from the definition. In case of $t \equiv f(t_1, \cdots, t_n) \prec_l f(s_1, \cdots, s_n) \equiv s$ and $[t_1, \cdots, t_n] \prec\!\prec_l [s_1, \cdots, s_n]$, there exists k such that $t_i \equiv s_i (i < k)$ and $t_k \prec_l s_k$. From induction hypothesis, we have $\neg(s_k \#_{\leq} t_k)$. It follows from $t_k \prec_l s_k$ that $\neg(s_k \preceq t_k)$ by (a) and transitivity of $\preceq_l$. Hence, we have $\neg(s \#_{\leq} t)$.

Finally, let's prove (c). We have $t \not\equiv \Omega$ and $s \not\equiv \Omega$ from $t \#_{\leq} s$. Letting $t \equiv f(t_1, \ldots, t_n)$ and $s \equiv g(s_1, \ldots, s_n)$, we also have $f = g$ from $\neg(s \#_{\leq} t)$. From these facts, there exists k such that

$$\forall j < k[t_j \preceq s_j] \wedge t_k \#_{\leq} s_k \tag{1}$$

and we have also

$$\forall i[\exists l < i[s_l \not\preceq t_l] \vee \neg(s_i \#_{\leq} t_i)]. \tag{2}$$

Then we have $\neg(s_k \#_{\leq} t_k)$, since $s_k \#_{\leq} t_k$ and (2) leads $\exists l < k[s_l \not\preceq t_l]$ contradicting to (1). Since $t_k \#_{<} s_k$, we have $t_k \prec_l s_k$ by induction hypothesis. On the other hand, we have $\forall j < k[t_j \preceq_l s_j]$ from (1) by (a). These conclude that $[t_1, \ldots, t_n] \prec\prec_l [s_1, \ldots, s_n]$, hence $t \prec_l s$. □

Lemma 4.10 *The relation $\#_{<}$ on T_Ω is acyclic.*

Proof By proposition 4.9. □

Proof of lemma 4.7 Let $X = \{q \in Red^+ \mid \forall p \in Red[p \#_{\leq} q]\}$ and $Y = Red^* - X$. Let $[p_1, \ldots, p_m]$ be a list sorted from Y topologically according to $\#_{<}$ in increasing order. Now let's prove that the list is standard. From construction of Y and the definition of left-incompatible TRS, it follows that

$$\forall p \in Y, r \in Red^*[p \neq r \Rightarrow (p \#_{\leq} r \vee r \#_{\leq} p)]. \quad (3)$$

Assuming $[p_1, \ldots, p_m]$ is not standard, the following two cases are possible:

$$\exists p_i, p_j \in Y[i < j \wedge \neg(p_i \#_{\leq} p_j)] \quad (4)$$

$$\exists p_i \in Y, \exists q \in X[\neg(p_i \#_{\leq} q)] \quad (5)$$

In case of (4), it follows from (3) that $p_j \#_{<} p_j$ which contradicts to the fact that the list is topologically sorted. In case of (5), we have $p_i \in Red^+$ from the definition of X. Since $p_i \notin X$, there exists $p_j \in Red$ such that $\neg(p_j \#_{\leq} p_i)$. Therefore we have $q \#_{<} p_i$ and $p_i \#_{<} p_j$ by the definition of left-incompatible. These lead $\neg(p_j \#_{\leq} q)$ by proposition 4.9(b),(c). Hence, we have $q \#_{<} p_j$, which contradicts $q \in X$.

The converse is trivial. □

By the proof of lemma 4.7, we can calculate a list of standard redex schemata $SRed$ of left-incompatible TRS.

Example 4.11 *Let* $Red = \{f(g(a,a,\Omega)), f(g(b,\Omega,a)), g(b,a,b), g(c,\Omega,\Omega)\}$. *One of $SRed$ of the system is*

$$[f(g(a,a,\Omega)), f(g(b,\Omega,a)), g(c,\Omega,\Omega), g(b,\Omega,a), g(b,a,b)].$$

Lemma 4.12 *SLI is a proper subclass of LI.*

Proof $SLI \subseteq LI$ is trivial by the definitions and lemma 4.7. $SLI \neq LI$ is shown from example 4.5.

Lemma 4.13 *Let R be a left-incompatible TRS with $SRed = [p_1, \ldots, p_n]$. Let $C[\ ]$ be a context such that $C[\Omega] \uparrow p_d$, $C[\Omega] \#_{\leq} p_i (1 \leq i < d)$ and $C[\Omega_{\{p_d\}}]$ display the leftmost direction for $\{p_d\}$. Then $C[\Omega_{TD}]$.*

Proof Since $C[\Omega] \#_{\leq} p_i (1 \leq i < d)$, we have $C[\Omega_{\{p_i\}}] (1 \leq i < d)$. From definition of standardness, it follows that $p_d \#_{\leq} p_j (d < j \leq n)$. Thus, by Lemma 4.2 we can show that $C[\Omega_{\{p_i\}}]$ for $p_i \in Red^*$. □

Lemma 4.14 *LI is a proper subclass of TS.*

Proof First, let's prove that each $t \in Red^{\prec}$ has a transitive direction. Letting $t \in Red^{\prec}$, there exists some $p_d \in SRed$ such that $t \# p_i (i < d)$ and $t \uparrow p_d$. Since $t \succeq p_d$ contradicts non-overlap property, we have $t \not\succeq p_d$. So t must have a direction for $\{p_d\}$, from the definition of the direction. By Lemma 4.13, the leftmost direction of t for $\{p_d\}$ is a transitive direction. By Lemma 3.12, we have $LI \subseteq TS$. $LI \neq TS$ is shown from the following system.

$$Red = \{f(\Omega, a, a), f(a, \Omega, b)\}$$

□

We introduce the notion of well-marked term[TSEP93].

Definition 4.15 *Let R be a TRS.*

- *$root(t)$ denotes outermost symbol of term t.*
- *Let $D = \{root(l) \mid l \to r \in R\}$ be the set of defined symbols. The set of marked symbols D^* is defined as $\{f^* \mid f \in D\}$ where each f^* is new symbol and has the same arity of f. The set of marked term is $T_{\Sigma \cup D^*}$, simply written by T^*.*
- *Let t be a marked term. $e(t)$ denotes the term obtained from t by erasing all marks. $\delta(t)$ denotes the Ω-term obtained from t by replacing all subterms satisfying $root(t) \in D$ with Ω. $\overline{\delta}(t)$ denotes $f(\delta(t_1), \ldots, \delta(t_n))$ where $t \equiv f(t_1, \ldots, t_n)$ $(0 \leq n)$.*

Definition 4.16 *$t \in T^*$ is well marked if*

$$\forall s \subseteq t[root(s) \in D^* \Rightarrow e(\delta(s)) \in NF_\Omega].$$

Lemma 4.17 *If $t \in T^*$ is well marked then $e(\delta(s)) \in NF_\Omega$ for any $s \subseteq t$.*

Proof Trivial □

Lemma 4.18 *Let $t \in T^*$ be well marked. If $root(t) \notin D$, $e(t)$ is strong head-normal form.*

Proof Since $root(t) \notin D$, we have $e(\delta(t)) \not\equiv \Omega$. It follows from $e(s) \succeq e(\delta(t))$ by lemma 3.5 and lemma 4.17 that $\omega(e(t)) \succeq \omega(e(\delta(t))) \equiv e(\delta(t)) \not\equiv \Omega$. □

Lemma 4.19 *Let $C[t]$ be a well-marked term. If $t \to s$ and s is well marked, then $C[s]$ is well marked.*

Proof It is enough to show that $e(\delta(C'[s])) \in NF_\Omega$ for any $C'[\,] \subseteq C[\,]$ such that $root(C'[s]) \in D^*$. Let $t \equiv C''[t'] \to C''[s'] \equiv s$. Since $C[t]$ is well marked and lemma 4.18, we have $e(\delta(C'[C''[\Omega]])) \equiv e(\delta(C'[C''[t]])) \in NF_\Omega$. Since s is well marked, we also have $e(\delta(s')) \in NF_\Omega$ by lemma 4.17. We can prove $e(\delta(C'[C''[s']])) \in NF_\Omega$ by using lemma 3.5. □

Definition 4.20 ([TSEP93]) *Let $p_d \in T_\Omega$ and let $t \equiv C[t_1, \ldots, t_k, \ldots, t_m] \in T^*$ and $\overline{\delta}(t) \equiv C[\Omega, \ldots, \Omega, \ldots, \Omega]$. Furthermore let $e(C)[\Omega, \ldots, \Omega_{\{p_d\}}, \ldots, \Omega]$ display the leftmost direction for $\{p_d\}$. Then we say that t_k is the leftmost directed subterm of t with respect to p_d.*

Definition 4.21 (Functional Strategy) *Let R be a left-incompatible TRS with $SRed = [p_1, \ldots, p_n]$. The procedure $HNF(t)$ computing the strong normal form of t is defined as follows.*

Input:

- *A well-marked term $t \in T^*$.*

Output:

- *A well-marked term t' such that $e(t')$ is a strong normal form of $e(t)$.*

Procedure $HNF(t)$:

(1) Find the first compatible pattern p_d to $e(\overline{\delta}(t))$ in the list $SRed$ if it exists; otherwise go to (5).

(2) If $e(\overline{\delta}(t)) \succeq p_d$ and $p_d \in Red^+$, go to (5)

(3) If $e(\overline{\delta}(t)) \succeq p_d$ and $p_d \in Red$, rewrite t to t' by the corresponding rule to p_d, and return $HNF(t')$.

(4) Let s be the leftmost directed subterm of t w.r.t. p_d. Replace s in t by $HNF(s)$ and let s' be the resulting term. Return $HNF(s')$.

(5) If $root(t) \in D$ then return $markroot(t)$, else return t, where $markroot(f(t_1, \ldots, t_n))$ denotes $f^*(t_1, \ldots, t_n))$.

Note that the matching should be done by ignoring the mark information of t in the reduction in step (2). However, we need not ignore the mark information in the substitution using in the reduction.

Lemma 4.22 *Let R be left-incompatible system and let $t \in T^*$ be well marked. If there exists a normal form of $e(t)$,*

i) $t \xrightarrow{*}_{TI} HNF(t)$,

ii) the procedure $HNF(t)$ terminates,

iii) $root(HNF(t))$ is in $(C \cup D^*)$, and

iv) $HNF(t)$ is well marked.

Proof Since $\Rightarrow$ is terminating by lemma 3.15, we can prove by induction on lexicographic composition of $(\to_{TI} \cup \supset_{TI})$ and number of unmarked symbol in t denoted by $|\, t \,|_{um}$.

- In case of the execution path (1)-(5), i),ii) and iii) is obvious. Since t is well marked, $\forall s \subseteq t[root(s) \in D^* \Rightarrow e(\delta(s)) \in NF_\Omega$. Since $HNF(t)$ may differ only root symbol from t and $e(\delta(HNF(t))) \equiv e(\overline{\delta}(t))$, we have to show that $e(\overline{\delta}(t)) \in NF_\Omega$ which is followed from condition in (1). Hence vi) is shown.

- In case of the execution path (1)-(2)-(5), the proof is almost the same as above. The differences are as follows: since $e(\overline{\delta}(t)) \succeq p_d \in Red^*$, there are no compatible $p_i \in Red$ to $e(\overline{\delta}(t))$. Hence, $e(\overline{\delta}(t)) \in NF_\Omega$.

- In case of the execution path (1)-(2)-(3), the reduction $t \to t'$ in (3) is obviously transitive index reduction of t. Therefore, i) - vi) is satisfied w.r.t. t' by induction hypothesis. Hence, i) - vi) hold clearly.

- In case of the execution path (1)-(2)-(3)-(4), Let $t \equiv C[s]$. $C[\Omega] \uparrow p_d$ and $C[\Omega]\#p_i (1 \leq i < d)$ from (1), and $C[\Omega_{\{p_d\}}$ displays the leftmost direction for $\{p_d\}$. Since $C[\Omega_{TD}]$ by lemma 4.13 and $C[z] \in NF_\Omega$, we have $C[\Omega_{TI}]$ by lemma 3.11. Hence $t \supset_{TI} s$. By induction hypothesis, i) - vi) is satisfied w.r.t. s. It follows that s' is well marked by lemma 4.19.
  - If $e(s) \equiv e(HNF(s))$, we have $\mid HNF(s) \mid_{um} < \mid s \mid_{um}$ because $root(s)$ is in D and $root(HNF(s))$ is not.
  - Otherwise, since $s \xrightarrow{+}_{TI} HNF(s)$, We have $t \equiv C[s] \xrightarrow{+}_{TI} C[HNF(s)] \equiv s'$ by proposition 3.14.

  In both of cases above, i) - vi) is satisfied w.r.t. s'. From these facts we can easily conclude i) - vi) w.r.t. t. □

Theorem 4.23 *Let R be left-incompatible TRS and let $t \in T$ have a normal form. The procedure $HNF(t)$ eventually terminates and term $t' \equiv e(HNF(t))$ is a head normal form of t.*

Proof Direct consequence of lemma 3.7, lemma 4.18 and lemma 4.22.

This theorem shows that we can compute head-normal form of given term t if it has normal form. The procedure to compute normal form of t is easily constructed by applying HNF to t in top-to-bottom order.

5 Conclusion

This paper extends the class LI, which is still proper sub-class of TS. It may be possible to make more extension of LI by introducing permutations of arguments for each defined function symbol. Although the problem to find the permutations and standard list of *Red* is interesting, the class is still proper sub-class of TS, which can be shown by following example:

$$\begin{aligned} Red = \{ \quad & f(c,a,a,\Omega), \quad f(c,b,\Omega,a), \\ & f(d,a,a,\Omega), \quad f(d,\Omega,b,a), \\ & f(e,a,\Omega,a), \quad f(e,\Omega,a,b) \quad \}. \end{aligned}$$

Acknowledgment

We would like to thank Professor Yoshihito Toyama for useful suggestion and also thank Professor Yasuyoshi Inagaki and Professor Toshiki Sakabe for the discussions on this work.

References

[Dav92] A. J. T. Davie. *An Introduction to Functional Programming Systems Using Haskell*, chapter 8. Cambridge University Press, 1992.

[DJ90] N. Dershouwitz and J.-P. Jouannaud. Rewrite Systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 6, pages 243–320, North-Holland, 1990.

[Dur94] I. Durand. Bounded, strongly sequential and forward-branching term rewriting systems. *J. Symbolic Computation*, volume 18, pages 319–352, 1994.

[Hen80] P. Henderson. *Functional Programming*, chapter 8. Prentice Hall International, 1980.

[HL79] G. Huet and J.-J. Levy. Call by need computations in non-ambiguous linear term rewriting systems. Technical Report 359, INRIA, 1979.

[Jon87] S. L. P. Jones *The Implementation of Functional Programming Languages*, chapter 11. Prentice Hall International, 1987.

[Klo92] J. W. Klop. Term rewriting systems. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume I. Oxford University Press, 1992.

[Str89] R. I. Strandh. Classes of equational programs that compile into efficient machine code. In *LNCS*, volume 355, pages 449–461. Springer-Verlag, 1989.

[TSEP93] Y. Toyama, S. Smetsers, M. van Eekelen, and R. Plasmeijer. The Functional Strategy and Transitive Term Rewriting Systems. In R. Sleep, R. Plasmeijer, and M. van Eelkelen, editors, *Term Graph Rewriting: Theory and Practice*, chapter 5, pages 61–75. John Wiley & Sons Ltd, 1993.

[Tur85] D. Turner. Miranda: A non-strict functional language with plimorphic types. In *LNCS*, volume 201, pages 1–16. Springer-Verlag, 1985.